AN EXACT ALGORITHM AND A LOCAL SEARCH HEURISTIC FOR A TWO RUNWAY SCHEDULING PROBLEM

A Thesis

by

AMRISH DEEP RAVIDAS

Submitted to the Office of Graduate Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

December 2010

Major Subject: Mechanical Engineering

AN EXACT ALGORITHM AND A LOCAL SEARCH HEURISTIC FOR A TWO RUNWAY SCHEDULING PROBLEM

A Thesis

by

AMRISH DEEP RAVIDAS

Submitted to the Office of Graduate Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

Approved by:

Co-Chairs of Committee,	Sivakumar Rathinam
	Darbha Swaroop
Committee Members,	Luca Quadrifoglio
Head of Department,	Dennis O'Neal

December 2010

Major Subject: Mechanical Engineering

ABSTRACT

An Exact Algorithm and a Local Search Heuristic for

a Two Runway Scheduling Problem. (December 2010)

Amrish Deep Ravidas, B.E., College of Engineering, Guindy - Anna University

Co–Chairs of Advisory Committee: Dr. Sivakumar Rathinam
Dr. Darbha Swaroop

A generalized dynamic programming based algorithm and a local search heuristic are used to solve the Two Runway Departure Scheduling Problem that arises at an airport. The objective of this work is to assign the departing aircraft to one of the runways and find a departing time for each aircraft so that the overall delay is minimized subject to the timing, safety, and the ordering constraints. A reduction in the overall delay of the departing aircraft at an airport can improve the airport surface operations and aircraft scheduling. The generalized dynamic programming algorithm is an exact algorithm, and it finds the optimal solution for the two runway scheduling problem. The performance of the generalized dynamic programming algorithm is assessed by comparing its running time with a published dynamic programming algorithm for the two runway scheduling problem. The results from the generalized dynamic programming algorithm show that this algorithm runs much faster than the dynamic programming algorithm. The local search heuristic with $k - exchange$ neighborhoods has a short running time in the order of seconds, and it finds an approximate solution. The performance of this heuristic is assessed based on the quality of the solution found by the heuristic and its running time. The results show that the solution found by the heuristic for a 25 aircraft problem has an average savings of approximately 15% in delays with respect to a first come-first served solution. Also, the solutions produced by a 3-opt heuristic for a 25 aircraft scheduling problem has

an average quality of 8% with respect to the optimal solution found by the generalized dynamic programming algorithm. The heuristic can be used for both real-time and fast-time simulations of airport surface operations, and it can also provide an upper limit for an exact algorithm. Aircraft arrival scheduling problems may also be addressed using the generalized dynamic programming algorithm and the local search heuristic with slight modification to the constraints.

ACKNOWLEDGMENTS

I thank Dr. Sivakumar Rathinam for his supervision, advice and guidance throughout my study at Texas A&M University; his contributions were integral to the completion of my research work and thesis. I thank my co-chair Dr. Darbha Swaroop and committee member Dr. Luca Quadrifoglio for their support throughout the course of my research.

I would also like to thank all my friends, department faculty and staff for making my time at Texas A&M a memorable experience. Finally, thanks to all my family members for their love, motivation and support

TABLE OF CONTENTS

CHAPTER Page

I INTRODUCTION 1

A. Problem Formulation 3
B. Thesis Organization 4

II LITERATURE REVIEW 5

III ALGORITHMS 9

A. Generalized Dynamic Programming Algorithm 9
B. Local Search Heuristic 12
1. K-Opt Heuristic Algorithm 12
a. Step 1 14
b. Step 2 14
c. Step 3 15
2. Remarks 15

IV SIMULATION RESULTS 16

A. Generalized Dynamic Programming Algorithm 17
B. Local Search Heuristic 19

V CONCLUSIONS 28

A. Summary 28
B. Conclusions 29
C. Future Work 29

REFERENCES 30

VITA 36

LIST OF TABLES

TABLE Page

I The separation required between the departure times of any two aircraft in seconds. 16

LIST OF FIGURES

FIGURE		Page
1	A solution in the 2-exchange neighborhood of Y	13
2	Average running time of GDP algorithm and Brentnall's Dynamic programming algorithm	18
3	Average running time of GDP algorithm and Brentnall's Dynamic programming algorithm for aircraft grouped to batches based on type	18
4	Average solution quality of the heuristics for a *searchspan* equal to 5	20
5	Average computation time of the heuristics for a *searchspan* equal to 5	21
6	Average solution quality of the 3-opt heuristic as a function of the *searchspan*	21
7	Average computation time of the 3-opt heuristic as a function of *searchspan*	22
8	Comparison of the solution quality of the 3-opt with respect to the FCFS algorithm	23
9	Comparison of the running times of the 3-opt with respect to the GDP algorithm	23
10	Average solution quality of the heuristics for a *searchspan* equal to 5 on input samples where each queue consisted of aircraft of same type	24
11	Average computation time of the heuristics for a *searchspan* equal to 5 on input samples where each queue consisted of aircraft of same type	25
12	Average solution quality of the 3-opt heuristic as a function of the *searchspan* on input samples where each queue consisted of aircraft of same type	25

FIGURE Page

13 Average computation time of the 3-opt heuristic as a function of *searchspan* on input samples where each queue consisted of aircraft of same type 26

14 Comparison of the solution quality of the 3-opt with respect to the FCFS algorithm on input samples where each queue consisted of aircraft of same type 26

15 Comparison of the running times of the 3-opt with respect to the GDP algorithm on input samples where each queue consisted of aircraft of same type 27

CHAPTER I

INTRODUCTION

Airport congestion is a crisis which many airports must overcome in the near future. According to Federal Aviation Administration forecast, the total number of U.S travelers is expected to increase at a rate of 2.7% per year. At this rate, it is predicted that the total number of U.S passengers using the air transportation system will increase from 689 million in 2007 to 1.1 billion or more in 2025 [1]. In addition, the U.S Congressional Joint Economic Committee [2] reported that in 2007 around 78% of the total delays of the aircraft occurred before takeoff in which 58% of delays occurred at the gates and the remaining 20% occurred when the aircraft were moving between gates and the runway. With the increase in the volume of air traffic over the past 20 years, these aircraft delays have also increased.

The delays at the airport gates and taxiways can be reduced by better aircraft scheduling and airport resource management. In order to reduce the aircraft delays and improve the efficiency of airport operations, research organizations like National Aeronautics and Space Administration (NASA) have been developing decision support tools for the airport. These automation tools aim to improve aircraft scheduling and airport resource management. Specifically, NASA is involved in development of both real-time automation tools such as Surface Management System [3] and fast-time tools such as the Surface Operations Scheduler [4] to schedule/predict the movements of aircraft at airports. The main objective of the research is to develop an exact algorithm and a heuristic to reduce the overall delay for a two-runway scheduling problem that can be a part of these decision support tools.

This thesis follows the style of *IEEE Transactions on Automatic Control*.

The Two Runway, Scheduling Problem (TRSP) aims to assign the aircraft to the runways and find a departing time for each aircraft such that the total delays of all the aircraft is minimized subject to the timing, safety, and chain-type precedence constraints for the aircraft. The timing constraint requires that the departing time of an aircraft at a runway is at least equal to its release time. The safety constraint states that the departing times of two successive aircraft on a runway should be separated by a minimum separation time in order for the trailing aircraft to avoid the wake vortices generated by the leading aircraft. This minimum separation time depends upon the type of the leading and the trailing aircraft. The precedence constraints require that the sequence of aircraft assigned to each runway satisfy a given chain-type ordering of the aircraft. A main reason for studying these chain-type precedence constraints is as follows: It is known that the TRSP without any precedence constraints can be equivalently formulated as a TRSP with chain-type precedence constraints where aircraft belonging to the same type are grouped together and a chain-type precedence constraint can be imposed within each group based on the release times of the aircraft. These chain-type precedence constraints provide a special structure to the problem which can then be exploited using algorithms based on dynamic programming. Another reason for studying scheduling problems with chain-type precedence constraints is that they also arise in other closely related two runway scheduling problems involving arrival aircraft as discussed in [5]. In the scenario addressed by Brentall [5], aircraft arriving at a given fix are considered to belong to a group and a chain-type ordering constraint is enforced based on the arrival time of the aircraft at the fix.

In this work, we first present an exact algorithm based on Generalized Dynamic Programming (GDP) approach to solve the TRSP. Then a fast, local search heuristic with k-exchange neighborhoods is presented for the same problem.

A. Problem Formulation

Consider a set of n departing aircraft divided into q queues. Let the number of aircraft in i^{th} queue, for $i = 1, ..., q$ be n_i. The aircraft in the i^{th} queue are denoted by $a_{i1}, a_{i2}, ..., a_{in_i}$. $\mathcal{A} := \bigcup_{i=1}^{r} \{a_{i1}, \cdots, a_{in_i}\}$ denotes the set of all departing aircraft. Let $\alpha(b)$ represent the release time of the aircraft b, *i.e.*, the earliest time an aircraft is available for departure. Let $t(b)$ be the decision variable which represents the departure time of aircraft b. The timing constraints requires that each aircraft must be scheduled after its release time, i.e., $t(b) \geq \alpha(b)$ for all $b \in \mathcal{A}$. Let $type(b)$ define the *type* of the aircraft and $type(b) \in \{type_1, type_2, ..., type_l\}$. For example, $type_1$ could correspond to a *small* aircraft, $type_2$ could correspond to a *large* aircraft etc. It is assumed that the two runways are independent, *i.e.*, the runways are spaced wide apart such that the departures at one runway does not affect the departing times of aircraft at the other runway. For two distinct aircraft $b_1, b_2 \in \mathcal{A}$, if b_2 departs after b_1 in the same runway, then a minimum separation time denoted by $sep(type(b_1), type(b_2))$ is required between their departure times of b_1 and b_2. This separation time depends only on the *type* of the leading and following aircraft. It is a safety measure imposed on an departing aircraft to avoid the turbulence created by an aircraft which had previously departed from the same runway. It is assumed that the separation times satisfies the triangular inequality, *i.e.*, $sep(type(b_1), type(b_2)) + sep(type(b_2), type(b_3)) \geq sep(type(b_1), type(b_3))\ \forall b_1, b_2, b_3 \in \mathcal{A}$. There is also a chain-type precedence constraint on all the aircraft belonging to the same queue. That is, for two distinct aircraft a_{ij} and a_{ik} belonging to the i^{th} queue, if a_{ij} and a_{ik} are assigned to the same runway and if $j < k$, then a_{ij} should depart before a_{ik}. The delay of an aircraft is defined as $t(b) - \alpha(b)$.

The objective of the Two Runway, Scheduling Problem (TRSP) is to assign every

aircraft $\in \mathcal{A}$ to one of the two runways and to find their departure times such that the total delays of all aircraft, $\sum_{b\in\mathcal{A}}(t(b) - \alpha(b))$ is minimized subject to timing, safety and the chain-type precedence constraints.

B. Thesis Organization

Chapter II provides a detailed literature review for the addressed two runway scheduling problem. The chapter also provides the motivation for the proposed Generalized Dynamic Programming approach.

Chapter III explains the Generalized Dynamic Programming based exact algorithm and a local search heuristic with $k-exchange$ neighborhood for the TRSP. The dynamic programming algorithm by Brentnall [5] and Psarftis [6] are also explained briefly in this chapter.

Chapter IV describes the simulation results of the GDP algorithm and the fast heuristic. To assess the performance of the GDP algorithm, its computation time is compared with a dynamic programming algorithm developed by Brentnall [7]. To corroborate the performance of the heuristic, we compare the average quality of the solution produced by the heuristic with respect to a First Come, First Served (FCFS) algorithm.

Chapter V summarizes and concludes this thesis. The conclusions are based on the results and discussions presented in Chapter IV.

CHAPTER II

LITERATURE REVIEW

Scheduling of aircraft at the runways has been of considerable interest to researchers in air traffic management for the past three decades. The basic aircraft scheduling problem involves finding optimal departing or arrival times of aircraft subject to all the operational constraints. Depending on the scenario, the geometry of the airport taxiways and the workload of the controllers, different constraints may be imposed on departing or arrival aircraft. In addition to these constraints, objectives may also differ based on the requirements and the scenario. Common objectives include minimizing the makespan of a sequence, minimizing total delays of aircraft etc.

Runways were identified as an important resource for scheduling arrivals and departures in Dear and Sherif [8],[9] and Idris et al. [10],[11] respectively. Several algorithms including exact algorithms and heuristics have been developed for different variants of the single runway, scheduling problem in [12]-[35]. Though majority of the literature have focused on single runway scheduling problems, there are several heuristics and exact algorithms available for Multiple Runway Problems (MRPs) also. In [36], Ciesielski and Scerri proposed a genetic algorithm for scheduling aircraft on two runways. Cheng et al. also develop genetic algorithms for a MRP in [37]. A hybrid approach combining a genetic algorithm with an ant colony heuristic has been developed for a MRP in [38]. Apart from genetic algorithms, Pinol and Beasley [39] have developed scatter search and bionomic heuristics for a MRP. Exact algorithms include formulating the scheduling problem as a mixed integer linear program and solving the resulting program using optimization software like CPLEX. Beasley et al. [12] formulate a mixed integer linear program for a MRP where the objective is to minimize the total deviation from the target landing time for each aircraft. They

solve the problem using a linear programming-tree based search. Ernst et al. [16] solve this MRP using a branch and bound method together with a specialized simplex method for finding the landing times of the aircraft. A novel branch and price method was also proposed by Wen et al. for the MRP in [17].

Dynamic programming has also been used for finding optimal solutions for aircraft scheduling problems whenever there are chain-type precedence constraints [5],[6] or there are constraints that restrict the takeoff position of an aircraft [6],[20],[21].

Psaraftis [6] was the first to develop a dynamic programming based exact algorithm for a single runway problem where aircraft are grouped based on their type and a chain-type precedence constraint can be enforced for all the aircraft within each group. The dynamic programming algorithm proposed by Psaraftis [6] uses $(k_1, k_2, ..., k_r, p)$ and $SPAN(k_1, k_2, ..., k_r, p)$ as the state and its corresponding optimal makespan of a partial schedule containing the first k_i aircraft of the i^{th} queue, where the last aircraft to depart in the schedule comes from the p^{th} queue. $SPAN(0, 0, ..., 0)$ is first set equal to 0 and the optimal makespan of all the other states are computed recursively as:

$$\begin{aligned} SPAN(k_1, ..., k_r, p) \;=\; & \min_{q=1,..,r:k'_q>0} [max(SPAN(k'_1, \ldots, k'_r, q) \\ & + sep(type(a_{qk'_q}), type(a_{pk_p})), \alpha(a_{pk_p}))], \end{aligned}$$

where

$$k'_i = \begin{cases} k_i - 1 & i = p \\ k_i & \text{otherwise.} \end{cases}$$
$$\text{for } i = 1 \cdots r$$

Even though this dynamic program proposed by Psaraftis works for the single runway problem, a direct extension of this dynamic program to either the single

runway or the two runway scheduling problem seems difficult if the objective is to minimize the total delay of all the aircraft. The reason for this is that the minimum total delay of a partial schedule corresponding to the state $(k_1, k_2, \cdots, k_r, p)$ not only depends on the minimum total delay corresponding to the state $(k'_1, \ldots, k'_r, q)$ but also on its makespan.

For this reason, Brentall [5],[7] proposed a dynamic programming algorithm for the total delay objective by defining $(k_1, k_2, \cdots, k_r, y_{11}, y_{12}, \cdots, y_{tt}, b, p)$ as the new state for the single runway scheduling problem where b denotes the *last* aircraft in the partial schedule that departed after an idle time period and preceded a sequence of y_{ij} number of type j aircraft following type i. An idle time period occurs when the runway is available but the aircraft that is next to depart from the runway is not available, *i.e.*, the departure time of the aircraft has not yet reached its release time. The implication of the definition of this new state is straightforward; the makespan of the partial schedule can now be easily calculated by using the information in the state about aircraft b and $y_{11}, \cdots, y_{tt}$, *i.e.*, the makespan of the state $(k_1, k_2, \cdots, k_r, y_{11}, y_{12}, \cdots, y_{tt}, b, p)$ is equal to $\alpha(b) + \sum_{i,j=1,\cdots,l} y_{ij} sep(type_i, type_j)$.

As given in Brentall[7], this basic idea can also be extended to a two runway problem by defining $(k_1, k_2, \cdots, k_r, y^1_{11}, y^1_{12}, \cdots, y^1_{tt}, y^2_{11}, y^2_{12}, \cdots, y^2_{tt}, b_1, b_2, p_1, p_2)$ as the state where p_k is the *last* departing aircraft at the k^{th} runway, b_k denotes the *last* departing aircraft at the k^{th} runway in the partial schedule that departed after an idle time period and preceded a sequence of y^k_{ij} number of type j aircraft following type i at the k^{th} runway. In this article, we take a new approach to address the TRSP. We retain the definition of the state $(k_1, \cdots, k_r, p_1, p_2)$, but however for each partial schedule associated with the state, we associate three objectives. The first objective, $DELAY_s(k_1, \cdots, k_r, p_1, p_2)$, denotes the total delay of the partial schedule s associated with the state $(k_1, \cdots, k_r, p_1, p_2)$. Similarly, $SPAN^1_s(k_1, \cdots, k_r, p_1, p_2)$

and $SPAN_s^2(k_1, \cdots, k_r, p_1, p_2)$ denotes the makespan of the sequences in the partial schedule s assigned to the first and second runway respectively. We then seek for all the pareto-optimal schedules corresponding to each state. A partial schedule corresponding to a state is *pareto optimal* or *non-inferior* if there exists no other partial schedule corresponding to the state that will yield an improvement in one objective without causing a deterioration in at least one of the other objectives. Basically, a partial schedule z is deleted only if it is inferior, $i.e$, there is another partial schedule, z', belonging to the same state as z such that:

- z' is better than or equal to z with respect to each of the objectives
- z' is strictly better than z with respect to at least one of the objectives

The optimal delay and a schedule corresponding to this optimum can then be obtained by finding a set of all the pareto optimal solutions corresponding to the state $(k_1, k_2, ..., k_r, p1, p2)$ and choosing a schedule that has the least total delay. Preliminary results of this approach for a single runway problem appeared in [28]. In this article, we present this algorithm for the TRSP and present computational results that show this approach is must faster than the Dynamic programming algorithm by Brentall [7].

CHAPTER III

ALGORITHMS

A. Generalized Dynamic Programming Algorithm

GDP algorithm is an exact algorithm which finds an optimal solution for the TRSP. A schedule y is represented by $y := (y_1, y_2)$ where y_1 and y_2 are the departing sequences of aircraft corresponding to the first and second runway respectively. If an aircraft $b \in \mathcal{A}$ is added as the last departing aircraft to a departing sequence, say y_1, then the new departing sequence y_1' is denoted as (y_1, b). A schedule is defined as a *partialschedule* if its departure sequences does not consist all the aircraft considered in the problem.

1. Let the set of all the partial schedules, S, initially be an empty set.

2. Let $S' := \emptyset$. For each partial schedule $y \in S$, do the following:

   - Let $F \subseteq \mathcal{A}$ be such that an aircraft $a \in F$ if and only if a is not present in the partial schedule y and a is the next aircraft waiting to depart from its queue given all the aircraft in y have departed.
   - For each $a \in F$, add a to the partial schedule y such that a is the last departing aircraft on the first runway. Let the new schedule $y' = (y_1, a), y_2$. Update S' with this new schedule, *i.e.*, $S' := S' \bigcup y'$. The departing time of aircraft a in this new schedule is the least available time that will satisfy all the safety and separation constraints. Specifically, since a is added to the first runway, its departure time in y' is defined as $\mathtt{T}(a, y') := \max(\mathtt{T}(b, y) +$

$sep(type(b), type(a)), \alpha(a))$ if b is the last aircraft to depart from the first runway in y and $\mathtt{T}(a, y') := \alpha(a)$ if there is no aircraft assigned to the first runway in y. The objectives of the new schedule $y' := \{(y_1, a), y_2\}$ are then equal to:

$$DELAY(y') = DELAY(y) + \mathtt{T}(a, y'), \tag{3.1}$$

$$SPAN_1(y') = \mathtt{T}(a, y'), \tag{3.2}$$

$$SPAN_2(y') = SPAN_2(y). \tag{3.3}$$

- Similarly, for each $a \in F$, add a to the partial schedule y such that a is the last departing aircraft on the second runway. Update S' with this new schedule, *i.e.*, $S' := S' \bigcup \{y_1, (y_2, a)\}$. As in the previous step, since a is added to the second runway, its departure time in y' is defined as $\mathtt{T}(a, y') := \max(\mathtt{T}(b, y) + sep(type(b), type(a)), \alpha(a))$ if b is the last aircraft to depart from the second runway in y and $\mathtt{T}(a, y') := \alpha(a)$ if there is no aircraft assigned to the second runway in y. The objectives of the new schedule $y' := \{y_1, (y_2, a)\}$ are then equal to:

$$DELAY(y') = DELAY(y) + \mathtt{T}(a, y'), \tag{3.4}$$

$$SPAN_1(y') = SPAN_1(y), \tag{3.5}$$

$$SPAN_2(y') = \mathtt{T}(a, y'). \tag{3.6}$$

3. Delete any partial schedule y' from S' if there exists another partial schedule $\overline{y} \in S'$ such that

   - $state(\overline{y}) = state(y')$

- $\overline{y}$ is better than or equal to y' with respect to each of the objectives and $\overline{y}$ is strictly better than y' with respect to at least one of the objective.

The partial schedules which are inferior are identified based on the Pareto Optimization technique discussed by H. T. Kung et al [40]. The algorithm to identify and remove an inferior partial schedule is as follows

(a) Let $Y := \{s_1, s_2, ..., s_l\}$ denote a set of l partial schedules corresponding to a state.

(b) Let $obj(s_i, 1) := DELAY(s_i)$, $obj(s_i, 2) := SPAN_1(s_i)$ and $obj(s_i, 3) := SPAN_2(s_i)$ where $i = 1, ..., l$.

(c) Sort the list of partial schedules Y such that following conditions are satisfied for every $s_i \in Y$

- $obj(s_{i+1}, 1) \geq obj(s_i, 1)$
- $obj(s_{i+1}, 2) \geq obj(s_i, 2)$ if $obj(s_{i+1}, 1) = obj(s_i, 1)$
- $obj(s_{i+1}, 3) \geq obj(s_i, 3)$ if $obj(s_{i+1}, 2) = obj(s_i, 2)$ and $obj(s_{i+1}, 1) = obj(s_i, 1)$

(d) Remove a partial schedule s_i if there exists another partial schedule $s_k \in Y$ for $k = 1, ..., i-1$ such that $obj(s_k, 2) \leq obj(s_i, 2)$ and $obj(s_k, 3) \leq obj(s_i, 3)$.

Let S_t be the new set of partial schedules obtained after removing all the inferior solutions form S'.

4. Let $S := S_t$ and repeat steps (2-3) $n - 1$ times, *i.e.* until each schedule in S consists of all the aircraft.

B. Local Search Heuristic

In this section, a local search heuristic with $k - exchange$ neighborhoods which runs fast and can find feasible solutions of good quality for the TRSP is explained. In this heuristic, a feasible solution is first generated using a simple First Come, First Served (FCFS) algorithm. Then, a new solution in the $k - exchange$ neighborhood of the current, feasible solution is chosen such that the new solution is feasible and has a lower delay. This procedure is iterated until a maximum number of iterations is reached or a given number of successive iterations do not lower the delay.

1. K-Opt Heuristic Algorithm

Some preliminaries and notations are presented before discussing the heuristic. A departing sequence for the i^{th} runway is denoted by $a_{\sigma(i,1)} \rightarrow a_{\sigma(i,2)} \rightarrow a_{\sigma(i,3)} ... a_{\sigma(i,p_i-1)} \rightarrow a_{\sigma(i,p_i)}$ where $a_{\sigma(i,j)} \neq a_{\sigma(i,j)}$ if $j \neq k$, p_i is the number of aircraft assigned to the i^{th} runway and $\sigma(i,j)$ denotes the index of aircraft present in the j^{th} takeoff position of the sequence assigned to the i^{th} runway. That is $a_{\sigma(i,1)}$ denotes the first departing aircraft at the i^{th} runway, $a_{\sigma(i,2)}$ denotes the second departing aircraft at the i^{th} runway and so on. The directed edge joining any two consecutive, departing aircraft in the representation of a sequence is such that the tail of each directed edge is incident on the leading aircraft.

The departing time of an aircraft in a given aircraft sequence is calculated based on the following simple rule: For each aircraft, chose the least available departing time that satisfies all the timing, safety, and the ordering constraints. To start with, the departing time of the first aircraft in the i^{th} runway, $t(a_{\sigma(i,1)})$, is chosen to be its release time $\alpha(a_{\sigma(i,1)})$. The departing time of the second aircraft in that runway, $t(a_{\sigma(i,2)})$, is determined by the timing constraints $t(a_{\sigma(i,2)}) \geq \alpha(a_{\sigma(i,2)})$, and the separation

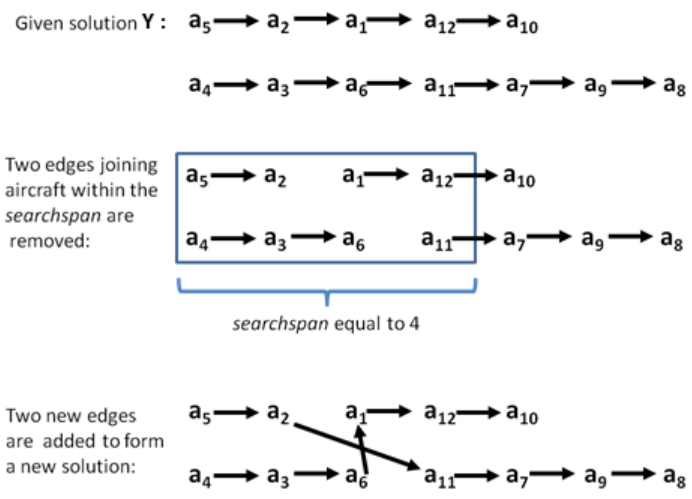

Fig. 1. A solution in the 2-exchange neighborhood of Y

constraints $t(a_{\sigma(i,2)}) \geq t(a_{\sigma(i,1)}) + sep(type(a_{\sigma(i,1)}), type(a_{\sigma(i,2)}))$.

The general expression for computing the best departing time for aircraft $a_{\sigma(i,j)}$ for $j \geq 2$ in the i^{th} runway is given by

$$t(a_{\sigma(i,j)}) = max(\alpha(a_{\sigma(i,j)}), t(a_{\sigma(i,j-1)}) + sep(type(a_{\sigma(i,j-1)}), type(a_{\sigma(i,j)})))$$

A feasible solution Y is represented by $y := (y_1, y_2)$ where y_1 and y_2 are the departing sequences of aircraft corresponding to first and the second runway. $DELAY(Y)$ represents the total delay of all aircraft corresponding to the feasible solution Y.

A solution Y_2 is said to be in the $k - exchange$ neighborhood of a solution Y_1 if Y_2 can be obtained from Y_1 by replacing k directed edges in Y_1 with k new directed edges. The $k - opt$ heuristic starts with a feasible solution Y and iteratively improves on this sequence through successive $k' - exchanges$ for any $k' \leq k$ until a maximum number of iterations is reached or a given number of successive iterations do not lower the delay. The $k - opt$ heuristic is explained below.

a. Step 1

The feasible solution Y obtained from a $FCFS$ algorithm is used as the initial solution for the $k - opt$ heuristic. Let the best solution found by the heuristic be denoted by Y_{best} . To start with $Y_{best} := Y$. Set the number of successive failed jumps, f_{max}, to be equal to 0. Also, set the number of iterations, n_{iter}, to be equal to 0.

b. Step 2

Set $index = 1$. Do the following until $index$ exceeds the length of both the runway sequences in Y:

I. Let $\mathcal{F} := Y$.

II. Form a set of all solutions denoted by $A_{\mathcal{F}}$ corresponding to $\mathcal{F}$ such that any solution $\mathcal{F}' \in A_{\mathcal{F}}$ must satisfy each of the following conditions

- $\mathcal{F}'$ is in the $k' - exchange$ neighborhood of $\mathcal{F}$ for some $k' \leq k$.
- Essentially to form $\mathcal{F}'$, $k' \leq k$ edges are removed from $\mathcal{F}$ and k' new directed edges are added. The edges which are removed from $\mathcal{F}$ should only be between aircraft whose takeoff position in their respective sequence is between $index$ to $index + searchspan - 1$. The head of an edge that is added can then only be incident on any aircraft whose takeoff position in $\mathcal{F}$ is in the range $index + 1$ to $index + searchspan - 1$. Refer to an illustration of these steps in Figure 1 for $index = 1$.
- $\mathcal{F}'$ is feasible with respect to all the ordering constraints.

III. Set $\mathcal{F} := \mathcal{F}^*$ where $\mathcal{F}^* \in A_{\mathcal{F}}$ and $DELAY(\mathcal{F}^*) \leq DELAY(\mathcal{F}') \; \forall \mathcal{F}' \in A_{\mathcal{F}}$.

IV. Set $index := index + searchspan - 1$. Return to step I if $index$ has not exceeded the length of both the runway sequences in Y.

c. Step 3

- $n_{iter} := n_{iter} + 1$;
- If $DELAY(Y) < DELAY(Y_{best})$ then set $Y_{best} := Y$.
- If $DELAY(F) \leq DELAY(Y)$ then set $f_{max} := f_{max} + 1$

If n_{iter} exceeds the given limit on the number of iterations or the number of successive failed jumps f_{max} has exceeded a given constant, terminate the algorithm and output Y_{best} as the best feasible solution found for the TRSP; Else, return to Step 2 of the heuristic.

2. Remarks

1. Varying the value of k in $k - opt$ would lead to a class of heuristics. In general, 2-opt and 3-opt are the main k-opt heuristics used in practice.
2. There are three parameters, $searchspan$, n_{iter} and f_{max}, that can be varied in this heuristic based on the scenario, desired solution quality and computation time available.

CHAPTER IV

SIMULATION RESULTS

Simulation results are presented in this chapter to asses the performance of the Exact GDP algorithm and the $k-opt$ heuristic. The algorithms were applied to the TRSP with 3 queues. The types of aircraft considered in the simulations were *small*, *large* and $B757$. The separation matrix given in table I was used for the simulations. For example, if a large aircraft departs after a small aircraft on the same runway, then the minimum separation between their departure times must be 42 seconds.

Table I. The separation required between the departure times of any two aircraft in seconds.

		Leading Aircraft Type		
		Small	Large	B757
Trailing Aircraft Type	Small	42	64	80
	Large	42	65	67
	B757	42	45	67

To simulate relatively congested traffic, the release time of each aircraft was uniformly chosen from the interval $[0, T(n)]$ where n is the number of aircraft and $T(n)$ is equal to $\frac{n\times 60}{2\times r}$secs. The number of aircraft in the simulation was varied from 8 to 25. For each number of aircraft n, 200 samples of $\mathcal{A}$ were generated. The aircraft type was assigned such that no single type was dominant. If $\mathcal{A}$ denotes the set n aircraft in a sample, then the first $\frac{n}{3}$ aircraft was set to type *small*, next $\frac{n}{3}$ were set to type *large* and rest were set to type $B757$.

The aircraft were assigned to a queue using the following rules.

- A random permutation for all the n aircraft from $\mathcal{A}$ was generated and the first $\frac{n}{3}$ aircraft were assigned to the first queue, next $\frac{n}{3}$ were assigned to the second queue and the rest to the third.
- The aircraft in each queue were sorted in the increasing order of their release times. For example, if aircraft in i^{th} queue have $8, 17\&5$ as their release times, then the aircraft were sorted such that $\{a_{i1}, a_{i2}, a_{i3}\}$ correspond to the aircraft with release times $\{5, 8, 17\}$.

All simulations were run on an Intel Xeon Quad Core E5540 CPU with 2.53GHz processing power and 12.0 GB RAM.

A. Generalized Dynamic Programming Algorithm

The performance of the Generalized Dynamic Programming algorithm is assessed by comparing its running time with the dynamic programming algorithm by Brentnall [7].

The results from Figure 2 show that the GDP algorithm runs much faster than the Brentnall's algorithm. For a 25 aircraft problem GDP algorithm on average found the optimal solution within 100 seconds. In comparison, the Brentnall algorithm on average took over 200 seconds to find the optimal solution for a 13 aircraft problem. For more than 13 aircraft, we could not find the solutions as the space needed to store the solutions found by Brentnall algorithm was more than the available memory resource on the computer.

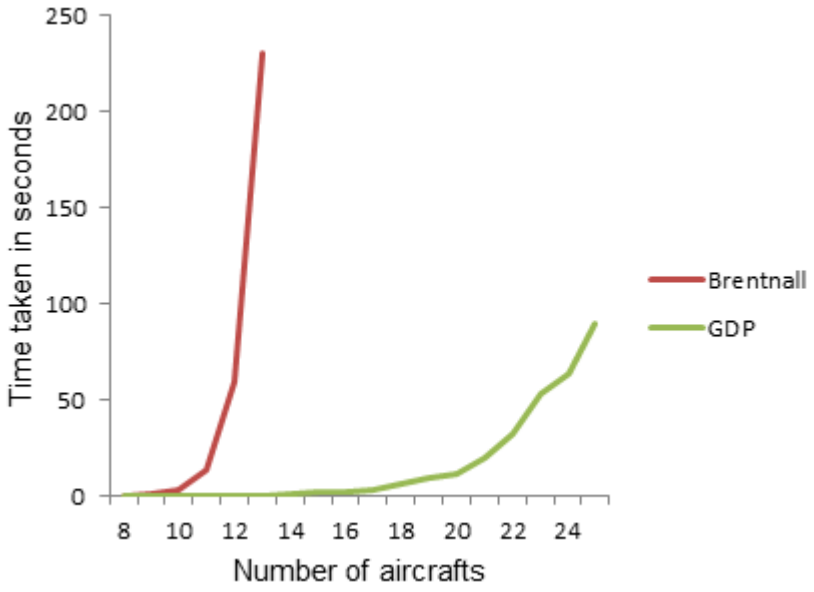

Fig. 2. Average running time of GDP algorithm and Brentnall's Dynamic programming algorithm

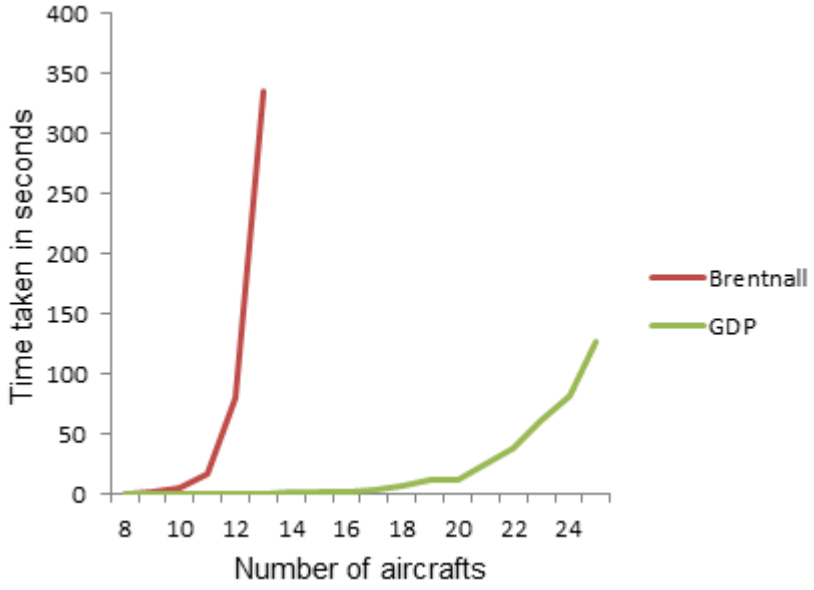

Fig. 3. Average running time of GDP algorithm and Brentnall's Dynamic programming algorithm for aircraft grouped to batches based on type

Figure 3 compares the running times of GDP and dynamic programming algorithm for samples where each queue consisted of aircraft having the same type. Aircraft of type *small* were assigned to the first queue, type *large* to second queue and type $B757$ to the third queue. The release times of aircraft were computed in the same way mentioned in the previous simulation and the aircraft in a queue were sorted in the increasing order of their release times. The motivation to group aircraft of similar types into a queue is as follows: It is known that the TRSP without any precedence constraints can be equivalently formulated as a TRSP with chain-type precedence constraints where aircraft belonging to the same type are grouped together and a chain-type precedence constraint can be imposed within each group based on the release times of the aircraft.

B. Local Search Heuristic

In this section, the results from the local search heuristic are presented. The performance of the heuristic is assessed based on the quality of the solution found by the heuristic and its running time. The quality of the solution produced by applying the $k-opt$ heuristic on an instance I is defined as

$$\frac{DELAY^{k-opt}(I)-DELAY^{*}(I)}{DELAY^{*}(I)} \times 100$$

where $DELAY^{k-opt}(I)$ is the total delay of the solution obtained by applying the $k-opt$ heuristic and $DELAY^{*}(I)$ is the optimal delay obtained through generalized dynamic programming.

In the first set of simulations, the parameters of the k-opt heuristic was chosen as follows: *searchspan* was equal to 5, the maximum number of successive failed jumps allowed was 1 and the number of maximum iterations was set at 50. Using these parameters, Figure 4 shows the average solution quality obtained using the 2-

opt, 3-opt and 4-opt heuristic as the number of aircraft increases. Figure 5 shows the average running times of the 2-opt, 3-opt and 4-opt heuristic. These results show that the 3-opt and 4-opt outperform 2-opt heuristic in terms of solution quality. Also, the average solution quality produced by 3-opt is approximately equal to that of 4-opt but the 3-opt heuristic on average runs 10 times faster than the 4-opt heuristic for a 20 aircraft problem. Therefore, 3-opt provides a good tradeoff between the solution quality obtained by a heuristic and its computation time.

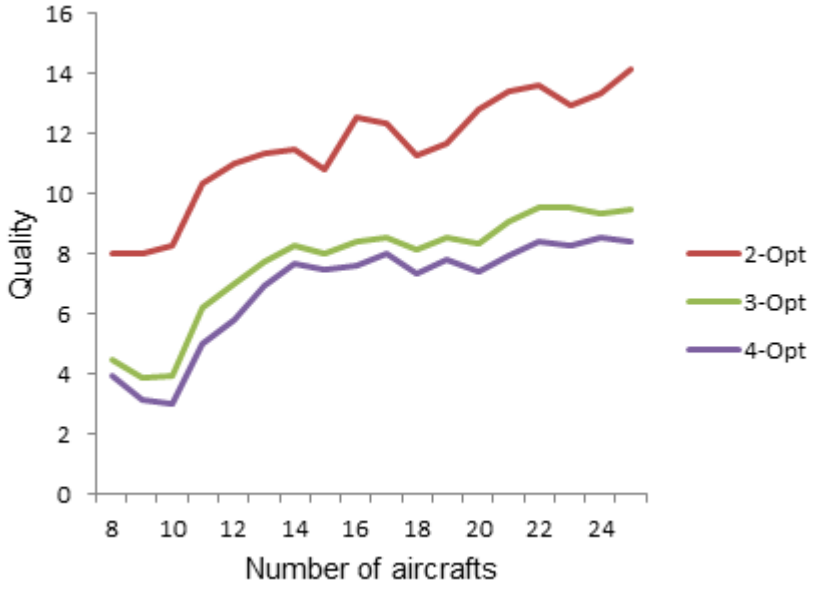

Fig. 4. Average solution quality of the heuristics for a *searchspan* equal to 5

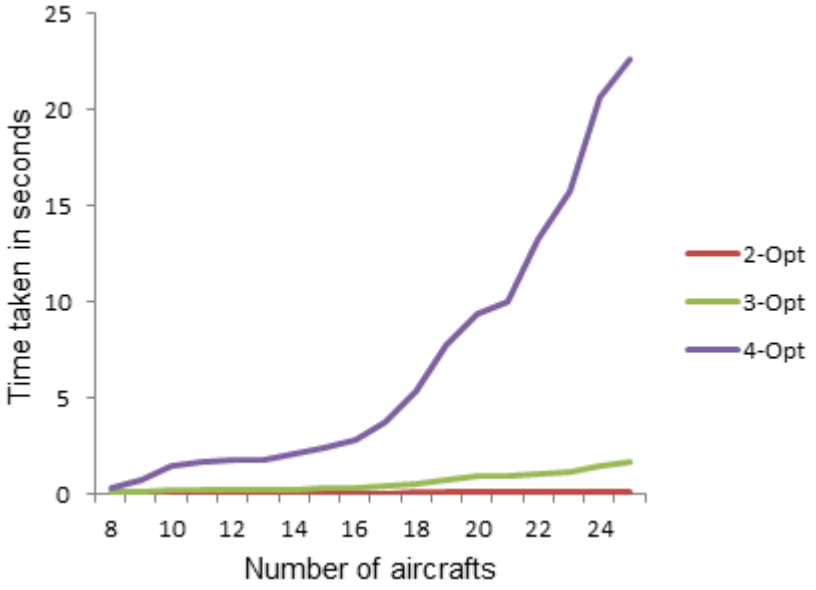

Fig. 5. Average computation time of the heuristics for a *searchspan* equal to 5

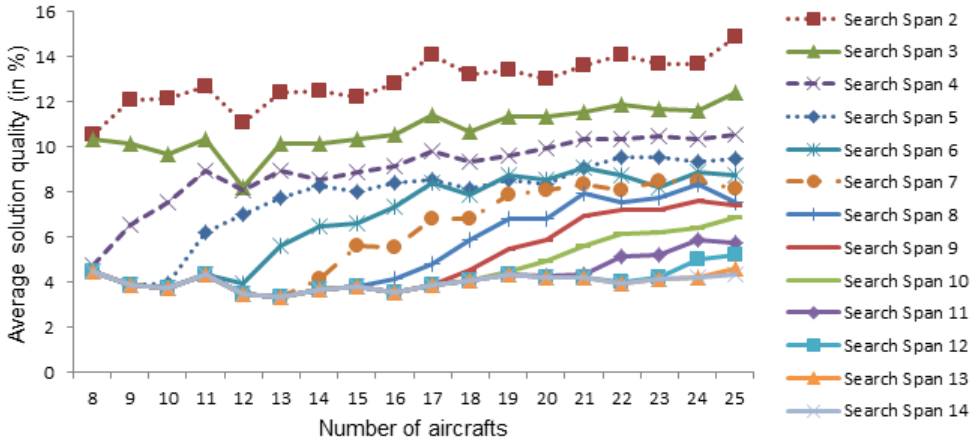

Fig. 6. Average solution quality of the 3-opt heuristic as a function of the *searchspan*

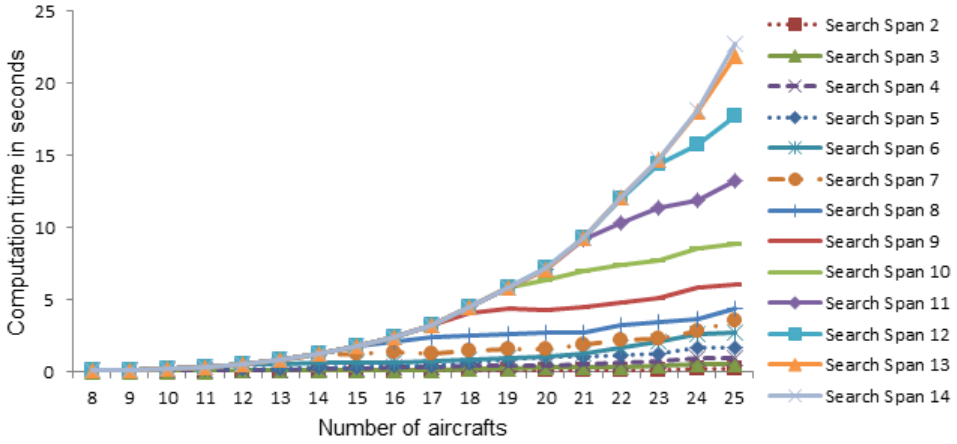

Fig. 7. Average computation time of the 3-opt heuristic as a function of *searchspan*

In the second set of simulations, *searchspan* was varied from 3 to 14 to understand the effect of *searchspan* on the solution quality and the running time of a 3-opt heuristic. The maximum number of successive failed jumps allowed was 1 and the number of maximum iterations was set at 50. For these simulations, the results on the solution quality and the computation time are shown in Figure 6 and Figure 7 respectively. Depending on the computation time available and the desired solution quality, one can use these two sets of simulation results as a guide to choose the right type of k-opt heuristic and the values for the control parameters. For a 25 aircraft problem, the 3-opt heuristic with a *searchspan* of 14 produced solutions with an average quality of at most 8% with running time in the order of seconds.

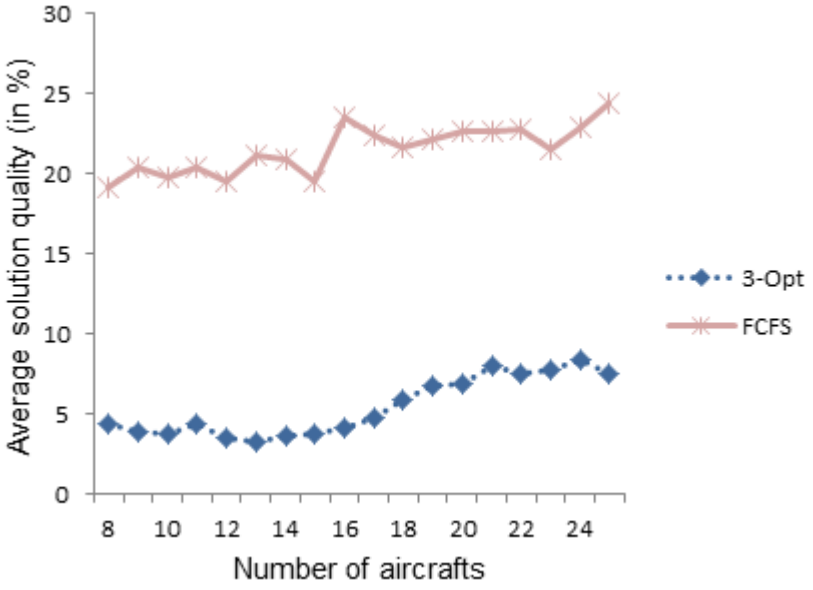

Fig. 8. Comparison of the solution quality of the 3-opt with respect to the FCFS algorithm

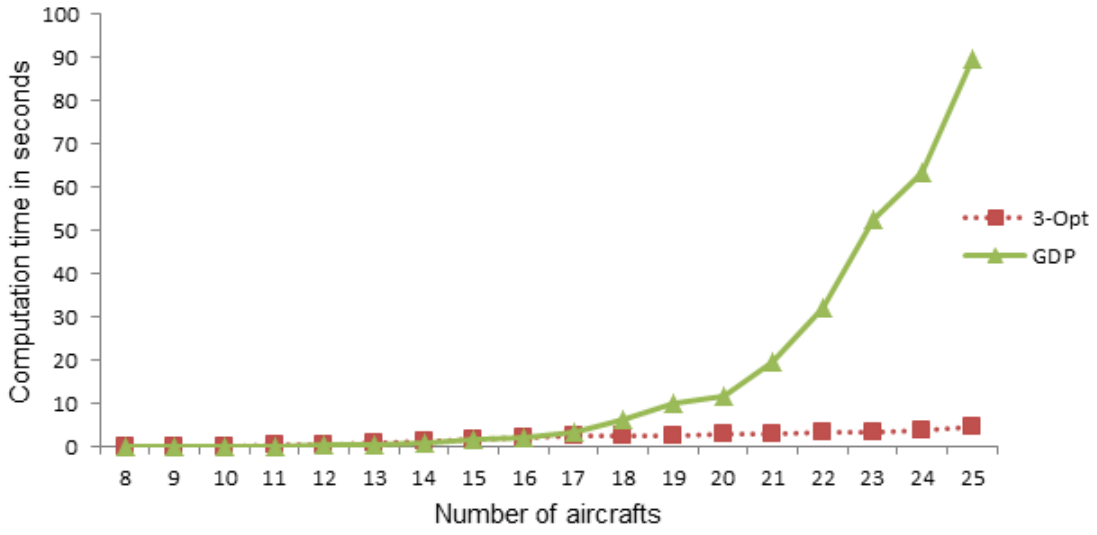

Fig. 9. Comparison of the running times of the 3-opt with respect to the GDP algorithm

To understand the average savings in the delays of the k-opt heuristic as compared to a FCFS solution, in Figure 8, both the average quality of the solutions found by the 3-opt heuristic with a search span of 8 is plotted alongside the average quality obtained using a FCFS algorithm. The results from Figure 7 in conjunction with Figure 8 show the following: With little effort, i.e., in approximately 4 secs, one can use the 3-opt heuristic to improve the quality of the solution produced by a FCFS heuristic by 15% for a 25 aircraft problem. Figure 9 compares the average running times of the GDP algorithm and the 3-Opt heuristic with a search span of 8, from the results we can see that the 3-Opt runs much faster than the GDP algorithm.

The simulations were repeated on input samples where each queue consisted of aircraft of same type. The results are presented in Figures 10-15

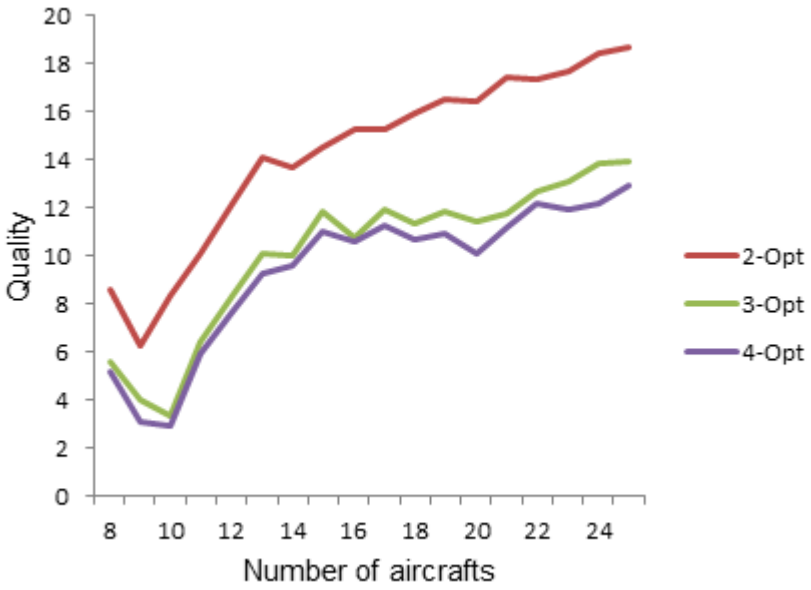

Fig. 10. Average solution quality of the heuristics for a *searchspan* equal to 5 on input samples where each queue consisted of aircraft of same type

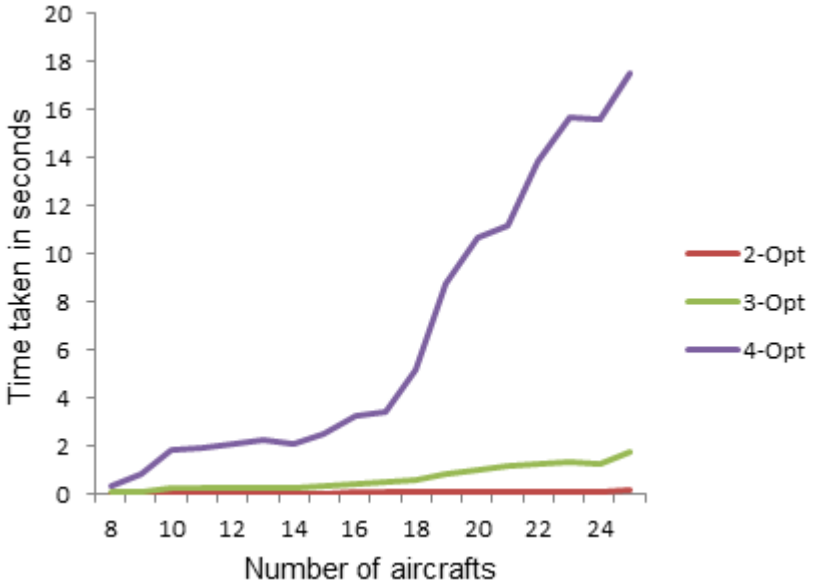

Fig. 11. Average computation time of the heuristics for a *searchspan* equal to 5 on input samples where each queue consisted of aircraft of same type

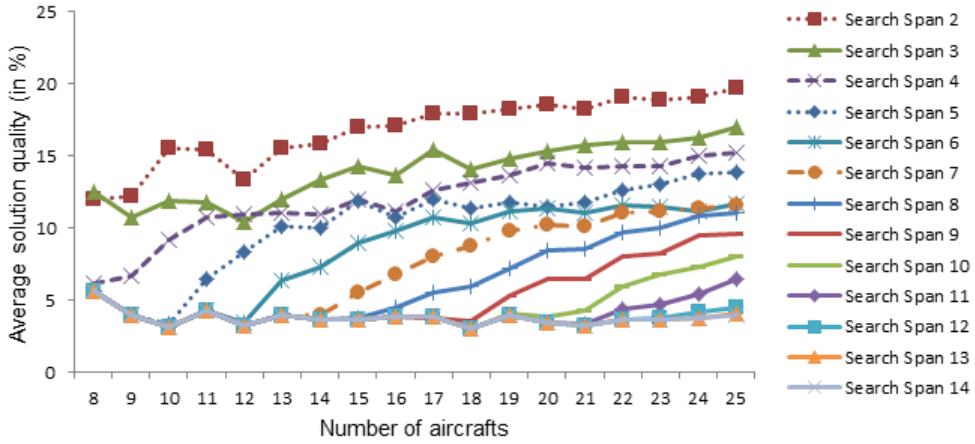

Fig. 12. Average solution quality of the 3-opt heuristic as a function of the *searchspan* on input samples where each queue consisted of aircraft of same type

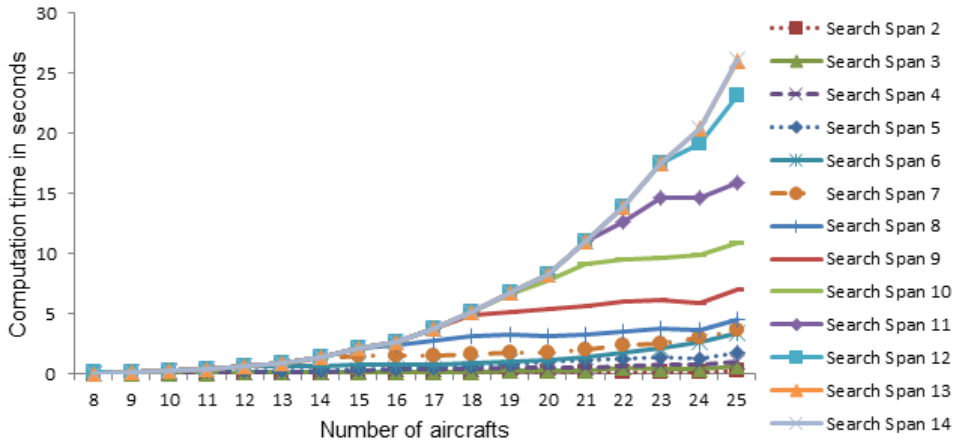

Fig. 13. Average computation time of the 3-opt heuristic as a function of *searchspan* on input samples where each queue consisted of aircraft of same type

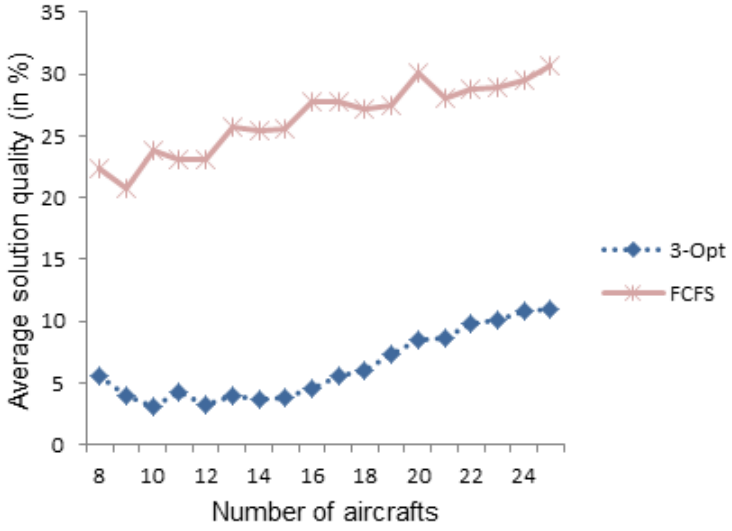

Fig. 14. Comparison of the solution quality of the 3-opt with respect to the FCFS algorithm on input samples where each queue consisted of aircraft of same type

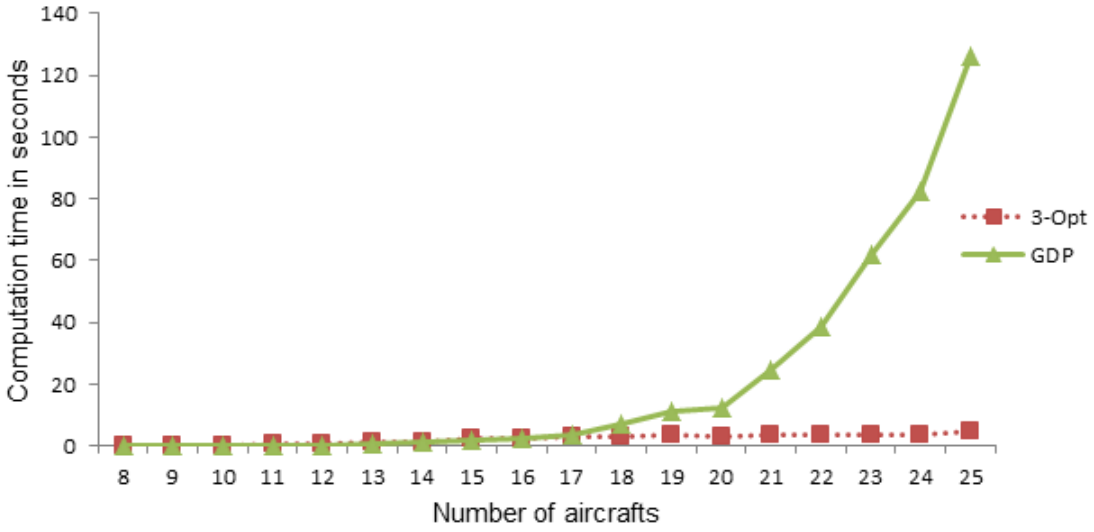

Fig. 15. Comparison of the running times of the 3-opt with respect to the GDP algorithm on input samples where each queue consisted of aircraft of same type

In the Figure 14, the solution quality of 3-opt search heuristic with a *searchspan* of 8 is compared with that of a FCFS algorithm for input samples in which the aircraft of similar types were grouped into queues. These results are similar to results obtained in the absence of explicit chain-type precedence constraints. The k-opt heuristic finds solution which are approximately 15% better in terms of quality than the FCFS algorithm for a 25 aircraft problem.

CHAPTER V

CONCLUSIONS

In this chapter, the results and key findings of this work are summarized. Based on these results, some important conclusions regarding the Generalized Dynamic Programming algorithm and the local search heuristic are presented.

A. Summary

GDP algorithm and K-Opt heuristic studied in this paper are promising algorithms which address the TRSP problems for decreasing the overall delay of the departing aircraft. Aircraft arrival scheduling problems can also be addressed using these algorithms with slight modifications to the constraints. The key results are summarized as follows.

1. The average running times of the GDP algorithm and DP algorithm by Brentnall [7] was compared. The results showed that GDP algorithm found optimal solution much faster than the DP algorithm by Brentnall [7].

2. The tradeoff between computation time and solution quality in selecting the optimal values for parameters like *searchspan* and $k-opt$ number for the local search heuristic was analyzed. Based on the analysis, a 3-opt algorithm with *searchspan* of 8 was selected.

3. The average solution quality of the local search heuristic was compared with that from the FCFS algorithm. The results showed that with minimal average computation time of about 4.5 seconds, the local search heuristic found solutions which were approximately 15% better in terms of quality than the FCFS algorithm for

a 25 aircraft problem.

B. Conclusions

Generalized dynamic programming algorithm and a novel search heuristic were developed for the two runway departure scheduling problem that arises at an airport. Computational results from the Generalized Dynamic Programming algorithm shows that it runs much faster than an existing Dynamic Programming algorithm developed by Brentnall [7]. This exact algorithm can be used to find an optimal solution to the Two Runway, Scheduling Problem.

Results found by the $k - opt$ heuristic for a 25 aircraft problem have an average savings of approximately 15% in delays with respect to a First Come, First Served solution. Moreover, the average running time of the heuristic is in the order of seconds. Also, the solutions produced by a 3-opt heuristic for a 25 aircraft scheduling problem has an average quality of 9.5% with respect to the optimal cost. The developed heuristic can be used for both real-time and fast-time simulations of surface operations at an airport. It can also be used to provide tight upper bounds for exact algorithm.

C. Future Work

The local search heuristic and GDP algorithm can be extended to accommodate the practical constraints like runway availability, runway preferences and closely spaced multiple runways. The heuristic and the GDP can be consolidated to provide a single hybrid algorithm in which the heuristic provide a strong upper bound for discarding the sub-optimal solutions from the GDP. This would greatly decrease the computation times of the GDP algorithm.

REFERENCES

[1] R. Schaufele. (2008, Apr.). "General Aviation Forecast Overview 2008-2025" presented to *MWCOG*, Washington. [Online]. Available: http://www.mwcog.org/uploads/committee-documents/kl5fV1xe20080409110604.ppt

[2] "Flight delays cost passengers, airlines and the U.S. economy billons," a report by the *U.S. Congress Joint Economic Committee Majority Staff*, May 2008.

[3] S. Atkins, Y. Jung, L. Stell, C. Brinton, and S. Rogowski, "Surface Management System Field Trial Results," *AIAA 4th Aircraft Technology, Integration, and Operations (ATIO) Forum*, Chicago, Illinois, September 20–22, 2004.

[4] Z. Wood, S. Rathinam, Y. Jung and M. Kistler, "A Simulator for Modeling Aircraft Surface Operations at Airports," presented at *AIAA Modeling and Simulation Technologies Conf.*, Chicago, Illinois, Aug. 2009.

[5] AR. Brentnall and RC. Cheng, "Some effects of aircraft arrival sequence algorithms," *Journal of the Operational Research Society*, vol. 60, no. 7, pp. 962–972, 2009.

[6] H.N. Psaraftis, "A dynamic programming approach for sequencing groups of identical jobs," *Operations Research*, vol. 28, no. 6, pp. 1347–1359, 1980.

[7] AR. Brentnall, "Aircraft arrival management," PhD dissertation, University of Southampton, Southampton, Hampshire, 2006.

[8] R.G. Dear and Y.S. Sherif, "The dynamic scheduling of aircraft in high density terminal areas," *Microelect Reliab.*, vol. 29, no. 5, pp. 743–749, 1989.

[9] R.G. Dear and Y.S. Sherif, "An algorithm for computer assisted sequencing and scheduling of terminal area operations," *Transportation Research Part A: General*, vol. 25, pp. 129–139, 1991.

[10] Idris, H. R., B. Delcaire, I. Anagnostakis, W. D. Hall, N. Pujet, E. Feron, R. J. Hansman, J.-P. Clarke, and A. Odoni, "Identification of flow constraint and control points in departure operations at airport systems," presented at *AIAA Guidance, Navigation Control Conf.*, Boston, MA, 1998.

[11] Idris, H. R., B. Delcaire, I. Anagnostakis, W. D. Hall, N. Pujet, E. Feron, R. J. Hansman, J.-P. Clarke and A. R. Odoni, "Observations of departure processes at Logan Airport to support the development of departure planning tools," in *Air Traffic Control Quart.*, vol. 7, no. 4, pp. 229–257, 1999.

[12] J.E. Beasley, M. Krishnamoorthy, Y.M. Sharaiha, and D. Abramson, "Scheduling aircraft landings – the static case," *Transportation Science*, vol. 34, no. 2, pp. 180–197, 2000.

[13] L. Bianco, P. Dell'Olmo and S. Giordani, "Minimizing total completion time subject to release dates and sequence-dependent processing times," *Annals of Operations Research*, vol. 86, pp. 393–415, 1999.

[14] L. Bianco, S. Ricciardelli, G. Rinaldi, and A. Sassano, "Scheduling tasks with sequence dependent processing times," *Naval Research Logistics*, vol. 35, no. 2, pp. 177–184, 1988.

[15] I.M. Ovacik and R. Uzsoy, "Rolling horizon algorithms for a single-machine dynamic scheduling problem with sequence-dependent setup times," *International Journal of Production Research*, vol. 32, no. 6, pp.1243–1263, 1994.

[16] A.T. Ernst, M. Krishnamoorthy, and R.H. Storer, "Heuristic and exact algorithms for scheduling aircraft landings," *Networks*, vol. 34, pp. 229–241, 1999.

[17] Min Wen, Jesper Larsen and Jens Clausen, "An Exact Algorithm for Aircraft Landing Problem," Technical Report, Informatics and Mathematical Modeling, Technical University of Denmark, DTU, September 2005.

[18] Reha Uzsoy, Chung-Yee Lee, and Louis A. Martin-Vega, "Scheduling semiconductor test operations: Minimizing maximum lateness and number of tardy jobs on a single machine," *Naval Research Logistics*, vol. 39, no. 3, pp. 369–388.

[19] Torsten Fahle, Rainer Feldmann, Silvia Götz, Sven Grothklags and Burkhard Monien, "The Aircraft Sequencing Problem," *Computer Science in Perspective: Essays Dedicated to Thomas Ottmann*, Springer-Verlag New York Inc., New York, pp. 152–166, 2003.

[20] H. Balakrishnan and B. Chandran, "Scheduling Aircraft Landings under Constrained Position Shifting," presented at *AIAA Guidance, Navigation and Control Conference*, Keystone, CO, August 2006.

[21] H. Lee and H. Balakrishnan, "Fuel cost, delay and throughput tradeoffs in runway scheduling," in *Proc. of the American Control Conference*, Seattle, WA, 2008, pp. 2449-2454.

[22] C.S. Venkatakrishnan, A. Barnett, and A.R. Odoni, "Landings at Logan airport: Describing and increasing airport capacity," *Transportation Science*, vol. 27, no. 3, pp. 211–227, 1993.

[23] J. Abela, D. Abramson, M. Krishnamoorthy, A. De Silva, and G. Mills, "Computing optimal schedules for landing aircraft," *The 12th National Conference of*

the Australian Society for Operations Research, Adelaide, July 7-9, 1993, pp. 71–90.

[24] I. Anagnostakis, H. R. Idris, J.-P. Clarke, E. Feron, R. J. Hansman, A. R. Odoni, and W. D. Hall, "A conceptual design of a departure planner decision aid," presented at *Third FAA/Eurocontrol Internat. Air Traffic Management R&D Seminar*, Naples, Italy, 2000.

[25] I. Anagnostakis, J-P. Clarke,D. Bhme and U. Vlckers, "Runway Operations planning and control: sequencing and scheduling," *AIAA Journal of Aircraft*, vol. 38, no.6, Nov.-Dec. 2001.

[26] I. Anagnostakis and J.-P. Clarke, "Runway operations planning, a two-stage heuristic algorithm," presented at *AIAA Aircraft, Tech., Integration Oper. Forum*, Los Angeles, CA, 2002.

[27] I. Anagnostakis, and J.-P. Clarke, "Runway operations planning: a two-stage solution methodology," *Thirty Sixth Hawaii Internat. Conf. System Sciences* (HICSS-36), Big Island, HI, pp. 79a, 2003.

[28] S. Rathinam, Z. Wood, B. Sridhar and Y. Jung, "A generalized dynamic programming approach to a departure scheduling problem," presented at *AIAA Conference on Guidance, Navigation and Control*, Chicago, Illinois, 2009.

[29] J.A.D. Atkin, E.K. Burke, J.S. Greenwood and D. Reeson, "A metaheuristic approach to aircraft departure scheduling at London Heathrow Airport," *Computer Aided Systems in Public Transport, Lecture Notes in Economics and Mathematical Systems*, vol. 600, (M.Hickman, P. Mirchandani, and S.Voss, Eds.) Springer, San Diego, CA, 2008.

[30] J.A.D. Atkin, E.K. Burke, J.S. Greenwood, and D. Reeson, "Hybrid metaheuristics to aid runway scheduling at London Heathrow Airport," *Transportation Science*, vol. 41, no. 1, pp. 90–106, 2007.

[31] J.A.D. Atkin, E.K. Burke, J.S. Greenwood and D. Reeson, "An examination of take-off scheduling constraints at London Heathrow Airport," in *Proc.* 10^{th} *International Conference on Computer-Aided Scheduling of Public Transport*, (CASPT2006), Leeds, UK, June 21–23, 2006, pp. 169–187.

[32] Paul Stiverson, "A study of heuristic approaches for runway scheduling for the Dallas-Fort Worth Airport," M.S. Thesis, Dept. Mechanical Engineering, Texas A & M University, College Station, May 2009.

[33] G. Gupta, W. Malik and Y. Jung, "A mixed integer linear program for airport departure scheduling," presented at 9^{th} *AIAA Aviation Technology, Integration, and Operations Conference* (ATIO), 2009.

[34] J.E. Beasley, M. Krishnamoorthy, Y.M. Sharaiha, and D. Abranmson, "Scheduling aircraft landings-the static case," *Transportation Science*, vol. 34, no. 2, pp. 180–197, 2000.

[35] J. Abela, D. Abramson,M. Krishnamoorthy, A. De Silva, and G. Mills, "Computing optimal schedules for landing aircraft," in *Proc. of the 12th National ASOR Conference*, pp. 71–90, 1993.

[36] V. Ciesielski and P. Scerri (1998), "Real Time Genetic Scheduling of Aircraft Landing Times," D.Fogel (ed.), in *Proc. IEEE International Conference on Evolutionary Computation* (ICEC98), IEEE, New York, USA, 1998, pp. 360–364.

[37] V.H.L. Cheng, L.S. Crawford, and P.K. Menon, "Air traffic control using genetic

search techniques," *IEEE International Conference on Control Applications*, Kohala Coast-Island of Hawaii, HI, 1999, pp. 643–649.

[38] G. Bencheikh, J. Boukachour, A. El Hilali Alaoui and F. El Khoukhi, "Hybrid method for aircraft landing scheduling based on a job shop formulation," *International Journal of Computer Science and Network Security*, vol. 9, no. 8, pp. 78–88, 2009.

[39] H. Pinol, and J.E. Beasley, "Scatter search and bionomic algorithms for the aircraft landing problem", *Scatter Search and Bionomic Algorithms*, vol. 171, no. 2, pp. 439–462, 2006.

[40] H. T. Kung, F. Luccio and F. P. Preparata, "On finding the maxima of a set of vectors," *Journal of ACM*, vol. 22, no. 4, pp. 469–476, 1975.

VITA

Amrish Deep Ravidas received his Bachelor of Engineering degree in mechanical engineering from College of Engineering Guindy, Anna University, India in 2005. He worked as a Software Engineer at Infosys Technologies Limited from 2005 to 2008. In the fall of 2008, he entered the graduate program in the Mechanical Engineering Department at Texas A&M University, College Station, and received his M.S. degree in 2010. In the spring 2009, he joined the research team under Dr. Sivakumar Rathinam. His research interests include optimization and control systems. He may be reached at the Department of Mechanical Engineering, 3123 TAMU, College Station, TX 77843. His email is amrishdeep@hotmail.com.

The typist for this thesis was Amrish Deep Ravidas.